Install Environment and Platform Specific Notes

Software installation is divided into three parts: PYSRC, EXSRC, and CDAT. The PYSRC distribution contains everything needed to make a Tcl/Tk Python executable that is current enough to run CDAT. (Note: if you already have a suitable Python installation, the PYSRC installation can be skipped. See details below in section 7.) The EXSRC distribution contains required external packages necessary to support CDAT. The CDAT distribution contains Python modules and subsystems that are explicitly designed to for the analysis of climate simulation and climate observed data.

GNU Make – GNU Project – Free Software Foundation (FSF)

Before installing CDAT, you must have GNU Make installed on your system. GNU Make is a tool which controls the generation of executables and other non−source files of a program from the program's source files. To obtain the latest version of GNU Make, please visit the web site: http://directory.fsf.org/make.html.

Where Should I Install CDAT?

The notation `<CDAT_INSTALL_DIRECTORY>` indicates the full path name of the directory into which CDAT is to be installed.

The notation `<CDAT_SRC_DIRECTORY>` indicates the full path name of the directory where you uncompressed and un−tarred the CDAT source code.

It is required that the parent directory `<CDAT_INSTALL_DIRECTORY>` exists, and that you have write permission in it. For distributions that will be used by others, for example `/usr/local/cdat-4.0` would be an appropriate name for the `<CDAT_INSTALL_DIRECTORY>`.

However, `/usr/local` is a poor choice, as chances of collisions with other installations and associated unintended consequences are too high and uninstalling becomes difficult. The `<CDAT_INSTALL_DIRECTORY>` will be created if necessary, with the following subdirectories: bin, DODS, lib, include, info, man, sample_data.

Platform Compiler Options

Most platforms will build CDAT, as−is with the GNU Compiler Collection (GCC) compiler, but on some platforms the environment variable "CC" may need to be set before building CDAT. See the table below for details.

Platform	Set CC environment variable	Notes
AIX	cc_r	
Cygwin	gcc	Use default gcc
HP−UX 11	cc +z	
IRIX 6.5	cc	Use configuration option "−c irix" to get n32 X11
Linux flavors	gcc	Use default gcc
Mac OS X	gcc	Use default gcc

OSF1	cc	
Solaris 8	cc –mt	
Solaris 9	gcc	Use default gcc

The way to set the environment variable CC depends on the shell used. For example:

csh or tcsh:

```
setenv CC value
```

bash or sh:

```
set CC=value; export CC
```

If the value is more than one word, double quotes must surround it. For example:

```
setenv CC "cc -mt"
```

Python Environment Variables

On all platforms you must unset the following Python environment variables, if they are set:

```
unset PYTHONPATH
unset PYTHONHOME
```

Special Macintosh OS X 10.3.x and 10.4.x Installation Instructions

Optional changing the shell mode (i.e., bash, tcsh, csh, etc.): Change your shell preference environment by typing: `chsh` at the prompt. Then edit the file that appears. (Follow instructions in the file for editing.) You then must reboot your machine for the change to take effect.

Setting 'root': To become the "su" root user in an xterm window, you must first set one menu item. To start, you will need to select the "Finder" icon on your desktop. In the "Finder" window, select the following icons: "Applications", then "Utilities", then "NetInfo Manager". In the main menu of the "NetInfo Manager" window and under the "Security" menu, select the "Authenticate" menu item and follow the instructions.

Note: If you don't remember the root password, you can put in the 1st Mac OS X install disk and change the password when the prompt message states that you should do so.

Note 2: Before that, or if you don't have the disk you can try to do the following:
select the "Finder" icon on your desktop. In the "Finder" window, select the following icons: "Applications", then "Utilities", then "NetInfo Manager". In the main menu of the "NetInfo Manager" window and under the "Security" menu, select the "Authenticate" menu item, "Disable Root", then try to "Enable Root" again it should complain about root password being empty, you can now reset it to your choice.

Xcode Tools Installation: If the –lcrt1.o or bundle1.o errors occur on compilation, then you need to reinstall the Xcode Tools! This is an unfortunate bug resulting from conflicting installers. The correct way to fix this problem is to install the full Xcode Tools 1.1 CD, which you can download from http://connect.apple.com/. (For testing the Mac OS X 10.3.x CDAT installation, we installed the latest Xcode Tools (version 1.5).)

Tiger Note: On Tiger you can also obtain Xcode Tools from http://connect.apple.com , we tested with version 2.1

You will also need the X11 libraries. You can only get this from the Xcode Tools 1.0 CD. From the CD, go to "Packages", then install "X11SDK.pkg". This will place the following libraries in your `/usr/X11R6/lib directory`:

libFS.a	libXTrap.a	libXi.a	libXt.a	libfntstubs.a
libGL.a	libXau.a	libXinerama.a	libXtst.a	libfontconfig.a
libGLU.a	libXaw.a	libXmu.a	libXv.a	libfontenc.a
libGLw.a	libXcursor.a	libXmuu.a	libXvMC.a	libfreetype.a
libICE.a	libXdmcp.a	libXp.a	libXxf86misc.a	liboldX.a
libOSMesa.a	libXext.a	libXpm.a	libXxf86vm.a	libpsres.a
libSM.a	libXfont.a	libXrandr.a	libdps.a	libxkbfile.a
libX11.a	libXfontcache.a	libXrender.a	libdpstk.a	libxkbui.a
libXRes.a	libXft.a	libXss.a	libexpat.a	

G77/G95 installation: Go to the http://hpc.sourceforge.net/ website and then get the g77v3.4−bin.tar.gz (Jaguar/Panther) binary or g95−tiger−bin.tar.gz (Tiger) Place the tar file in the "/" directory. Become root and issue the command:

```
tar xvfz g77v3.4-bin.tar.gz
```

or

```
tar xvfz g95-bin.tar.gz
```

This will put the binaries in `/usr/local/bin`. (The g77/gfortran installation is only needed for some "contrib" modules and is not necessary for CDAT building.)

CDAT installation: Make sure your xterm window has g77/gfortran by typing:

```
which g77
```

or

```
which gfortran
```

If not found, then follow appropriate the steps above.

Finally, follow the standard Express Installation instructions above.

Problem with "readline" module on Mac OS X 10.4.x

On Tiger you might not be able to build python with readline, since Python seems to systematically pick the system readline which is not complete.

To solve this issue, simply remove readline from your /usr/lib and build the system again. You may want to relink from /usr/lib to your readline.

Fink CDAT Installation on Mac OS X 10.3.x

From the Fink website,"Fink is defined as a project that wants to bring the full world of Unix Open Source software to Darwin and Mac OS X. It has two main goals:

- First, to modify existing Open Source software so that it will compile and run on Mac OS X.
- Second, to make the results available to casual users as a coherent, comfortable distribution that matches what Linux users are used to."

The project offers precompiled binary packages as well as a fully automated build−from−source system.

To install "fink", follow the instructions located at the following website:

- http://fink.sourceforge.net

By the courtesy of Jeff Whitaker, CDAT can be installed with the Fink application by typing:

```
fink install cdat
```

Other useful Linux/Unix applications can be installed by using Fink. For a full listing of applications, type:

```
fink list
```

Fink CDAT Installation on Mac OS X 10.4.x (Tiger)

Special thansk to Kevin Marsh for these

```
1) (re)install fink (using 0.8)

2) "sudo gcc_select 4.0"

3) "fink install cdat"
-complains about missing c++-filt in /usr/bin/ when trying to build
gcc-3.4.3

4) reinstall bsd.pkg from the Tiger DVD (in
"/System/Installation/Packages" folder, not "Xcode Tools") to get
/usr/bin/c++filt. Actually locating and installing the pesky missing
c++-filt was eventually done with Jeff's help and hunting through a few
newsgroup postings (see
http://comments.gmane.org/gmane.os.apple.fink.general/17670). It looks
like c++filt is not always reinstalled after the Tiger upgrade.

5) "fink install cdat"
-finishes, but complains "can't build cdms"; cdat & vcdat are created in
/sw/bin/

6)sudo gcc_select 3.3

7) "fink install cdat"
-reports "unpacking cdat-4.0", but still no cdms. Tried
removing/reinstalling cdat several times, but no cdms built.
```

```
8) "fink purge cdat"

9) rebuild cdat (under fink commander)

10) "fink install cdat" (still using gcc3.3)
-this completes ok; I can now get a vcdat gui and plot some test Netcdf
data.
```

Special Cygwin (Windows XP) installation instructions

Install Cygwin from http://cygwin.com. When selecting the Packages for the Cygwin installation, it is important to select "Install" for "Base" and "X11". This will ensure that all the appropriate libraries needed for CDAT builds properly. In general, it is not a bad idea to select "Install" for "All" packages if you have the disk space.

Note: The "Cygwin Setup" can be a little confusing for first time users. When the package selection window appears, you will see the package name, a cycle icon, and the text "Default". If you select the text "Default" it will change to "Install". At the very top of this window you will see the "All" line. For the "All" line, we recommend that you toggle the selection from "Default" to "Install". This toggle from "Default" to "Install" will also change all subsequent package preinstall indications from "Default" to "Install" and will insure that the packages needed for CDAT will be installed accordingly.

Once you've installed Cygwin, select the Cygwin icon (located on your desktop) or from the "Start Menu" select "All Programs" then "Cygwin Bash Shell". In the window that appears, type:

```
rebaseall
```

If your TMP or TEMP environment variables are not set properly, you may need to issue the following command:

```
TMP=/tmp rebaseall
```

After issuing the `rebaseall` command, type:

```
startx
```

Note: It is extremely important to issue the `rebaseall` command; otherwise threading for Python and CDAT will not work properly.

Note: If your `$HOME` path has a space (or spaces) in it, then you will need to remove the space(s) by editing the `/etc/passwd` file. Make sure your account has "Computer Administrator" privileges. If not, then you will need to log on with an account that does. In the `/etc/passwd` file, find your user account and change your home directory path to remove all spaces. For example, if your path has:

- `/home/firstname lastname:/bin/bash`

change it to:

- `/home/lastname:/bin/tcsh`

When you have finished making your changes, you are required to reboot your machine for the changes to occur.

Special Ubuntu Linux installation instructions

Special thansk to Johnny Lin and Alexis Zubrow for this contribution.

Introduction

In early Feb 2006 I did the following express install at the Unix command line:

```
$ sudo ./express_install /usr/local/cdat-4.0 --enable-ioapi --disable-opendap
```

of CDAT 4.0 on a four−processor Dell Poweredge 6300 running Ubuntu 5.10 Server with the symmetric multi−processing (smp) kernel. This works, but there are some fixes you have to apply along the way.

Explicitly set the default compiler

If you don't explicitly set the environment variables specifying which compilers to use, you may end up compiling different libraries with different compilers. Thus, in csh or tcsh, make sure you execute:

```
setenv CC gcc
setenv FC g77
setenv CXX g++
```

or in Bash:

```
export CC=gcc
export FC=g77
export CXX=g++
```

Do this before you run the express install.

But I want OpenDAP!

If you want OpenDAP then leave out the `--disable-opendap` keyword in the above `express_install` call. However, installation with OpenDAP often fails, and since I didn't need OpenDAP, I decided to try the install with it disabled.

If at first you don't succeed ...

Throughout the install process, different problems might come up (packages aren't found, things don't build, etc.). In general, if that happens, my advice is to just delete everything in the target directory and the source directory, unpack the `tar` file again, and reinstall. (The recommended way is to just clean up the build directory, but I like the brute force approach to cleaning.) Sometimes this is enough to fix the problem.

If reinstalling doesn't work, then you'll have to look at the log files. (Btw, the logs are in `exsrc/build` and are named `<package-name>.LOG`. You might also be interested in the `config.log` files which are located in the package sub-directories in `exsrc/build`.)

Ubuntu packages needed

If you have only the minimal Ubuntu install, there is a good chance that you won't be able to do a successful install of CDAT, since additional packages are needed. Here's the list of what I added (you can add them using Synaptic, Ubuntu's GUI-based package manager):

- `bison`
- `byacc`
- `flex`
- `gawk`
- `g++`
- `g77`
- `libxt-dev`
- `tcsh`
- `tk8.4-dev`
- `xlibs-dev`

Of these packages, `xlibs-dev` is the really key one (see, for instance, Alexis Zubrow's experience with installing on Debian, which is very similar to Ubuntu). You may be able to do a successful build without adding all of the others, but because I didn't have the time to try reinstalling by adding one package at a time, I don't know if all the above is necessary. But with them all, it works.

gplot

First, in order to compile `gplot`, make sure you have the `libxt-dev` and `xlibs-dev` libraries installed. It won't work without them.

The next parts are tricky. You have to edit the `Makefile.Linux` file in `<CDAT_SRC_DIRECTORY>/exsrc/src/gplot`, where you compile `gplot`. The problem is that the X Windows libraries, which are normally in `/usr/X11R6/lib` are in `/usr/lib` in Ubuntu. But it gets worse. A number of the libraries in `/usr/lib` are not found, not because they are not present, but because they are named by version number. As an example, do `ls /usr/lib/libXp.*`. You'll find there are only two files, `libXp.so.6` and `libXp.so.6.2.1`. The `libXp.so` file is missing, which will cause the `gplot` compilation to break.

To prevent this, I created softlinks for all the "missing" libraries that are given in line 87 of the makefile:

```
MLIBS = -L/usr/X11R6/lib -lXp -lXpm -lXaw -lXmu -lXext -lXt -lX11
```

Thus, I created a `libXp.so` file by soft linking it to `libXp.so.6.2.1`, and so on for all the libraries specified in the line above that did not exist in `/usr/lib`. (Actually, to be more precise, I made a copy of all the libraries in `/usr/lib`, created the softlinks, and changed the `-L` flag in the makefile to that new directory. I did this because I didn't want to make changes to `/usr/lib`.

With these changes, `gplot` compiles fine, and you can following the rest of the instructions for installing `gplot`.